An Interop Publication

ConneXions®

The Interoperability Report

July 1992 Volume 6, No. 7

ConneXions — The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published monthly by Interop Company, 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. 415-941-3399. Fax: 415-949-1779. Toll-free: 1-800-INTEROP.

E-mail: connexions@interop.com.

ISSN 0894–5926

From the Editor

Last month I talked about "neat new stuff"—applications developed for use in the Internet (or in private internets) that go far beyond the basic functionality of e-mail, file transfer and remote login. This month, we bring you articles on no less than three such systems:

Prospero is a distributed directory service and file system that allows users to construct customized views of available resources while taking advantage of the structure imposed by others. It provides the framework in which various indexing services can be tied together, producing the fabric upon which other resource discovery techniques can operate. Prospero is based on the *Virtual System Model,* where users construct their own virtual systems by selecting objects and services that are available over the network.

The *Internet Gopher* is a simple client/server protocol designed to be used for building distributed information systems and organizing access to Internet resources. Gopher has become quite popular, and is now used at over 100 sites worldwide to distribute information including *news,* genome databases, Q&A about computer systems, and collections of Telnet-accessible library catalogs.

The *WorldWideWeb* (WWW) project merges the techniques of information retrieval and hypertext to make an easy but powerful global information system. It aims to allow information sharing within internationally dispersed teams, and the dissemination of information by support groups.

According to Michael Edelman, author of this month's *Opinion* article, well-managed networks are the exception rather than the rule. His conclusion was reached by evaluating many network management products over a period of 18 months. These opinion pieces seem to have taken a life of their own, and we encourage readers to contribute writings for future editions.

Also in this issue is the report from INTEROP 92 Spring. Personally, my high points of the week were *The Great OSI Debate,* and the presentation of the first *INTEROP Genesis Award* to Van Jacobson of Lawrence Berkeley Laboratory. Van has made a number of extraordinary contributions to the field of internetworking. His brilliant work on round-trip time estimation and flow control in TCP has dramatically improved TCP's ability to work over a wide-range of network conditions. His work on improving protocol implementations has demonstrated that both TCP and IP can perform at gigabit speeds and he has assisted efforts to get these improvements widely deployed. In addition, Van has made notable contributions to the fields of protocol compression over low-speed phone lines and to the design of high-speed network interfaces.

Prospero: A Virtual Directory Service for the Internet

by B. Clifford Neuman, Information Sciences Institute,
University of Southern California

Overview

Recent growth of the Internet has increased the amount of information that is accessible and the number of resources that are available to users. To exploit this growth, it must be possible for users to find the information and resources they need. Existing techniques are inadequate when searching for information on a global scale.

Prospero is a distributed directory service and file system that allows users to construct customized views of available resources while taking advantage of the structure imposed by others. It provides the framework within which various indexing services can be tied together, producing the fabric upon which other resource discovery techniques can operate.

Introduction

This article describes the *Prospero File System.* Prospero is based on the *Virtual System Model,* a model for organizing large systems within which users construct their own virtual systems by selecting objects and services that are available over the network; users then treat the selected resources as a single system, ignoring those resources that were not selected. Prospero supports customized views of a global file system, making it easier for users to keep track of files that they have identified as being of interest.

Tools are provided that help users organize their name spaces, allowing views to be defined as functions of other, possibly changing, views. These tools improve the user's ability to organize information, making it easier for users to identify information of interest than it is in existing file systems.

The Prospero File System is heterogeneous; instead of providing its own methods for storing and accessing files, it relies in existing file systems for storage and supports multiple underlying access methods. Prospero is implemented as a distributed directory service that names individual files, plus a file system interface that calls the appropriate access method once a name has been resolved. The prototype supports Sun's *Network File System,* the *Andrew File System,* and the *File Transfer Protocol* (FTP). For FTP, the file is automatically retrieved and the locally-cached copy is then opened.

A prototype is running and has been used from more than 7,500 systems in 29 countries on six continents. Experience with this prototype has shown that the organizational flexibility provided by the Virtual System Model is useful. Initial observations have provided insight into the way that users organize and look for information when they are not constrained to use a single, monolithic name space.

The benefits of customization

Instead of forcing all users to share a single hierarchy, Prospero allows users to create customized views of the global file system. This allows information that is of interest to a user to be prominently located near the center of the user's name space, while information that is not of interest can be kept out of the way. In contrast, systems supporting a single hierarchy, X.500 [1] and the *Andrew File System* [2] in particular, manage globally shared names by assigning organization names near the top of the tree, and often the names of users at the next level. These systems do little to help the user find files of interest; files on particular topics are scattered across the leaves of the tree, where they are difficult to find.

The customization supported by Prospero is important for a number of reasons: it reduces the clutter that would otherwise be caused by files in which the user has little interest; it allows users to define shorter names for frequently referenced files; and it allows users to replace entire portions of the naming hierarchy with alternative views more appropriate for their particular needs. Customization is particularly important in a system as large as the Internet because there are many communities of users, and they don't share the same interests.

Diversity of interest presents problems for systems supporting a single hierarchical name space. Where descriptive names are needed it is difficult to gain consensus on what subjects should appear near the top of the tree, and once topics are agreed upon, there is disagreement on what should be included under each topic. This problem is apparent on USENET, a worldwide distributed message service for disseminating messages on many topics. A significant share of the messages sent on USENET discuss what messages are appropriate for particular newsgroups, whether new newsgroups should be created, and what they should be called. Customization eliminates the need for consensus: each user can make a decision based on his or her own opinions.

One way that information is found using Prospero is by browsing. An individual interested in a particular topic can connect to the virtual system of someone known to be interested in that topic, and then look for documents or files of interest. Customization can improve the effectiveness of browsing by encouraging users to make their own links to the files in which they have an interest. As such, interesting files are likely to appear in the hierarchies of many people, thus increasing the likelihood that the files will be found by browsing.

Support for customization

As users organize virtual systems for their own use, the structure imposed on the information can often be used by others. The Prospero naming network forms a generalized directed graph. A user's name space appears hierarchical and corresponds to the names seen by the user starting from a particular node in the graph, the root of the name space. If a user finds an object or a directory of interest, the user can add a link that will make the object more prominent. When a user creates a directory for a particular topic, others can (if authorized) view that directory and include it in their own virtual systems, thus benefiting from the organization imposed by the first user.

Prospero makes directory information available from many sources, including automated indexing services and information specified by users. By treating the results of a query as a virtual directory, indexing services can make a large base of information available through Prospero. Users can make links to the resulting directories. In this manner, users combine such information in useful ways as they incorporate it into their own virtual systems.

Filters and links

Two features of Prospero, the *filter* and the *union link,* allow new directories to be derived from directory information that already exists.

When constructing a virtual system, a user can attach a filter to a link. A filter affects the way a directory is viewed, creating a derivative view from views that already exist.

continued on next page

Prospero *(continued)*

For example, Figure 1 shows files named with the labels *a* through *g*. Associated with each file is an attribute list, one attribute of which is the language in which the text of the file was written. The value of the language attribute is shown in the box representing the file. By attaching the *distribute()* filter to the directory link, a derivative view is created within which the files appear to be distributed across subdirectories according to the value of the language attribute. The derivative view is shown in Figure 2.

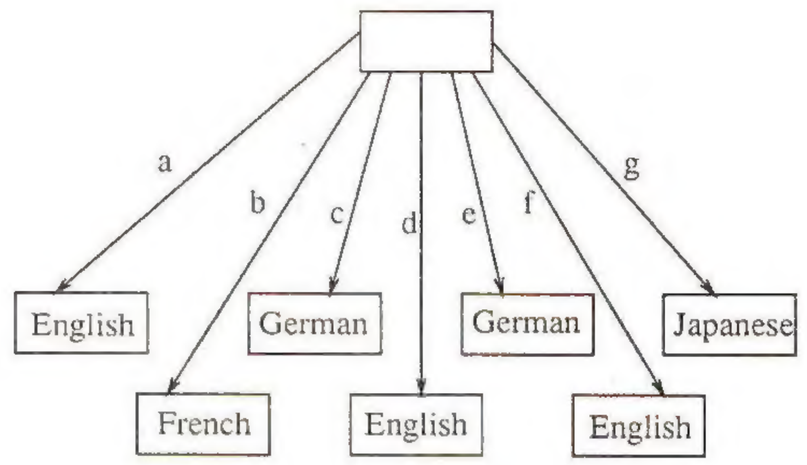


Figure 1: Directory before application of a filter

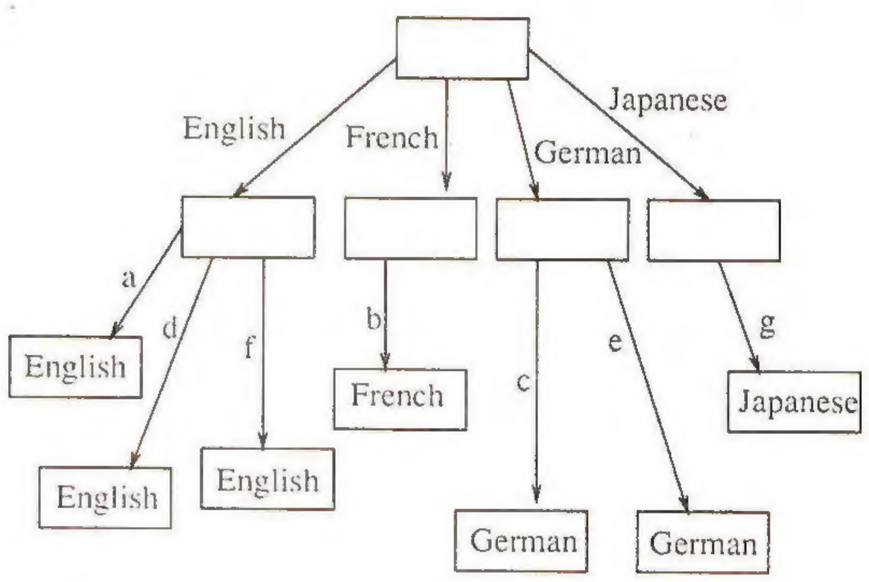


Figure 2: Directory with *distribute()* applied.

A filter can be an arbitrary program. It takes a representation of a directory as an argument and returns the same. It can add links to a directory, delete links, change the names of links, and even define new filters that are to be applied when traversing links deeper in the hierarchy. As arbitrary programs, filters can access any information needed to perform their function. Typically this information includes attributes of files and the contents of other directories, but it might involve reading files or performing database queries. Though users can write their own filters, it is expected that most will use the set already defined for them.

If a union link is included in a directory, the contents of the directory that is the target of the link appear to be included in the directory containing the link. This allows a directory to incorporate directory information from other sources. When the original source changes, the changes will also be reflected in the directory incorporating that information.

Customization does not come without cost. In traditional systems, a custom name space would cause confusion since the same name might refer to different objects at different times. Prospero avoids this problem by supporting closure: every object has an associated name space, and names specified by the object are resolved in that name space.

Over time, multiple communities of users will evolve. It is expected that the members of each community will have similarly structured name spaces, but name spaces may vary widely across different communities of users. For example, members of the computer science community might organize virtual systems in one way while members of the medical community might think of the world in a completely different manner.

If individuals do not like the way information is organized, they can organize it themselves, or they can find different experts whose views more closely match their own. They can completely customize their own name space so that their alternative view is used instead of the more accepted view. In fact, which view is the accepted view becomes more a matter of whose views more people adopt, rather than officially sanctioned.

The architecture

Filenames are resolved by contacting directory servers on the hosts that store Prospero directories. The server accepts the system level name of a directory and optionally the name of the link to be returned. The server returns the links in the directory that match the specified name, or all links if the name was not specified.

Attributes are associated with objects and the directory server responds to requests for specific attributes. To access an object a message is sent to the directory server requesting the value of the *access-method* attribute. The response includes a list of acceptable access methods together with the information needed to access the object using each.

The client remembers the current working directory and the root of the active virtual system. When a user or application wishes to resolve a name, the first component is resolved relative to the appropriate directory by sending a query to the corresponding directory server. The next component is resolved by sending a query to the directory server named in the link returned by the first query.

This process is repeated until all components of the name have been resolved. An optimization allows a directory server to resolve more than one component of the name at a time as long as all intervening directories are stored on the same server.

If at any point a union link is returned, the current component of the name is resolved in the directory identified by the union link and the result is merged with the current directory. If a filter is encountered the directory is read and the filter applied before the current component of the name is looked up. Filters are written in C and are dynamically linked with the name resolver when they are applied.

The client interacts with the directory server using a reliable communication protocol implemented on top of UDP. This reduces the overhead incurred when establishing connections to multiple directory servers.

Figure 3, on the following page, shows a client resolving a name using Prospero. The client exchanges messages with a file server, then with a server maintaining an *archie* database, and finally with another file server. Prospero makes these potentially diverse sources of information appear as a single system.

continued on next page

Prospero *(continued)*

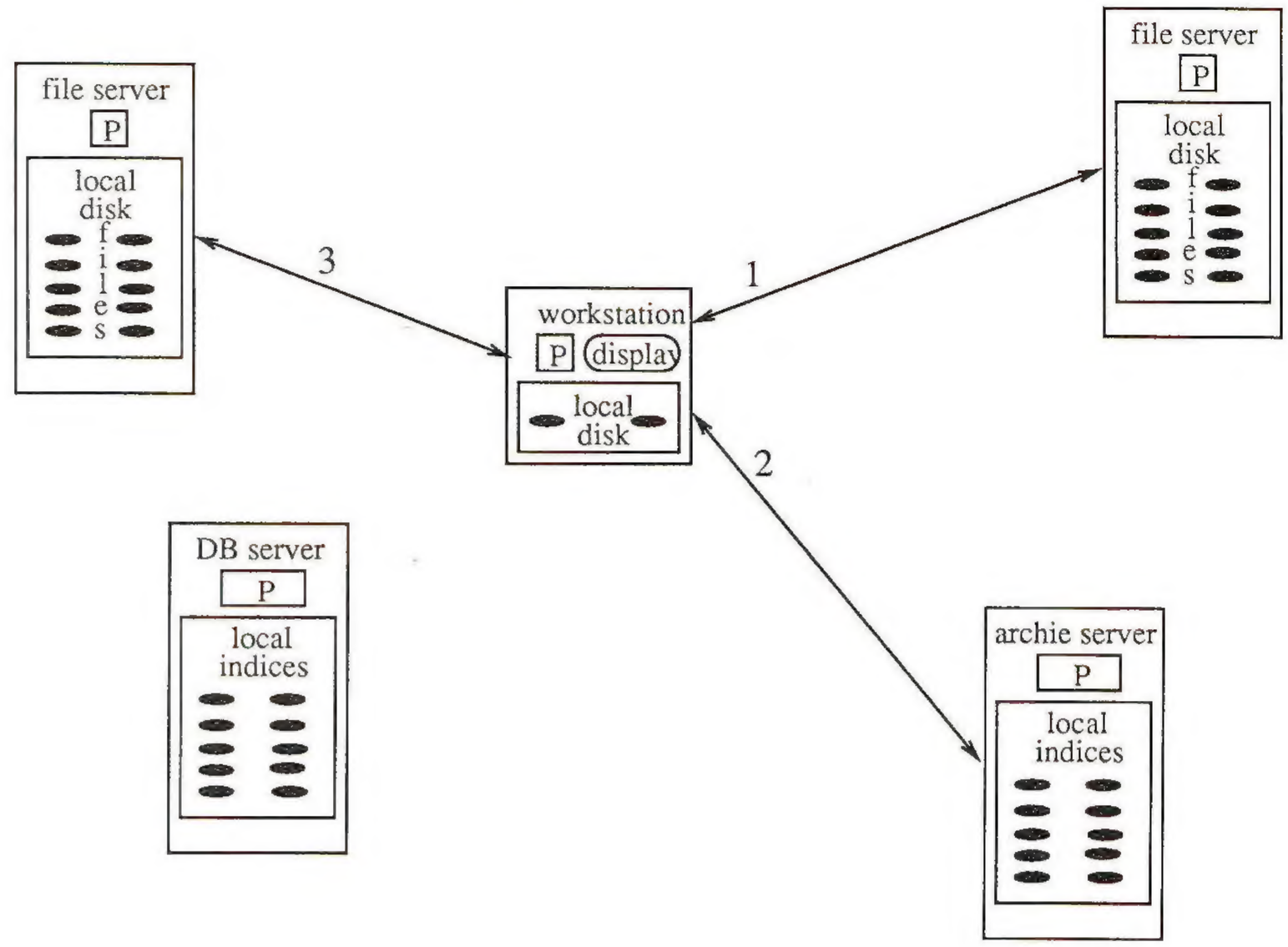


Figure 3: Resolving a name using Prospero.

Prospero allows access control lists to be associated with directories or individual links in directories. Such authorization attributes apply to the ability to resolve names in or to modify the directory, not to the ability to access the referenced object itself. Access control for the referenced object depends on the underlying access method, either NFS, AFS, or FTP.

Using Prospero

Prospero can be used in several forms. When used as a distributed file system, applications linked with the Prospero library open files they normally would, but the file names are resolved using Prospero. Separate utilities are provided for navigating and modifying the name space.

Figure 4 shows a sample session with Prospero. Users find information by moving from directory to directory in much the same manner as they would in a traditional file system. Users do not need to know where the information is physically stored. In fact, the files and directories shown in the example are scattered across the Internet. At any point, a user can access files in a virtual system as if they were stored on his or her local system. In the example, the user connects to the "root" directory and lists it using the *ls* command. The result shows the categories of information included in this virtual system. The information includes online copies of papers (in the "papers" directory), archives of Internet and USENET mailing lists (in the "mailing-list" and "newsgroups" directories), releases of software packages (in the "releases" directory), and the contents of prominent Internet archive sites (in the "sites" directory). Files of interest can appear under more than one directory. For example, a paper that is available from a prominent archive site might also be listed under the "papers" directory.

Next, the user connects to the papers directory, lists it, and finds the available papers further categorized as conference papers, journal papers, or technical reports. The "technical report" directory is broken down by organization, and by department within the organization.

The "journals" directory is organized by the journal in which a paper appears, and the two journals that are shown are further organized by issue. Use of the *vls* command shows where a file or directory is physically stored, demonstrating the fact that the files are scattered across the Internet (IEEE TC/OS Newsletter on `FTP.CSE.UCSC.EDU` and *Computer Communications Review* on `NNSC.NSF.NET`.) Though not shown in the example, papers are also organized by author and subject in other directories from the same virtual system.

```
% cd /
% ls
afs                info               papers
databases          lib                projects
documents          mailing-lists      releases
guest              newsgroups         sites
% cd papers
% ls
authors            conferences        subjects
bibliographies     journals           technical-reports
% cd technical-reports
% ls
Berkeley     IAState       OregonSt     UCalgary    UWashington
BostonU      MIT           Purdue       UColorado   Virginia
Chorus       NYU           Rochester    UFlorida    WashingtonU
Columbia     NatInstHealth Toronto      UKentucky
Digital      OregonGrad    UCSantaCruz  UMichigan
% ls UCSantaCruz
crl
% ls UCSantaCruz/crl
ABSTRACTS.1988-89           ucsc-crl-91-01.ps.Z
ABSTRACTS.1990              ucsc-crl-91-02.part1.ps.Z
ABSTRACTS.1991              ucsc-crl-91-02.part2.ps.Z
ABSTRACTS.1992              ucsc-crl-91-02.ps.Z
INDEX                       ucsc-crl-91-03.ps.Z
ucsc-crl-88-28.ps.Z         ucsc-crl-91-06.ps.Z
...
% ls UWashington
cs   cse
%
% ls UWashington/cs
1991           INDEX          PRE-1991
1992           OVERALL-INDEX  README
% cd /papers
% ls
authors            conferences        subjects
bibliographies     journals           technical-reports
% ls journals
acm-sigcomm-ccr  ieee-tcos-nl
% ls journals/ieee-tcos-nl
app-form.ps.Z  v5n1            v5n3
cfp            v5n2            v5n4
% ls journals/acm-sigcomm-ccr
application.ps  jan89           jul90           sigcomm90-reg.ps
apr89           jan90           oct88
apr90           jan91           oct89
apr91           jul89           sigcomm90-prog.ps
% vls journals
   acm-sigcomm-ccr       NNSC.NSF.NET      /usr/ftp/CCR
   ieee-tcos-nl          FTP.CSE.UCSC.EDU  /home/ftp/pub/tcos
%
```

Figure 4: Sample session using Prospero

It is important to note that the example shows only part of the information available through Prospero, and that it shows a typical way that the information is organized. Individuals can organize their own virtual systems differently.

Some of the most frequently queried directories are those that are part of the *archie* database, developed at McGill University [3]. Directories from the archie database organize files according to the last components of their file names. For example, the subdirectory "prosp" contains references to files available by Anonymous FTP whose names include the string "prosp." Among the matches would be files related to Prospero.

Prospero can be used as a directory service, independent of the file system interface, and it is in this manner that application specific clients, such as the widely available archie clients, access Prospero directories and indices. In this form, applications call the Prospero library to resolve names and query directories in the global naming network. The results are returned in a linked list of matches, possibly with accompanying attributes. The applications can then use the results in an application specific manner.

continued on next page

Prospero *(continued)*

There is also an FTP interface to Prospero that has been developed by Steve Cliffe of the Australian Academic and Research Network (AARNet) Archive Working Group. Prospero support has been added to one of AARNet's FTP servers. As well as making files available from the physical file system, the modified FTP server makes files available from a virtual file system. When a retrieval request is received, the FTP server locates the file using Prospero and checks to see if a copy of the file is available locally. Using Prospero to check the last modified time of the authoritative copy, the FTP server checks that the local copy is current. If a current copy does not exist locally, the server retrieves and caches a copy of the file. The local copy is then returned to the client.

Future plans

Prospero is an evolving system. We are continuing to work closely with the *archie* group to make additional databases available. Immediate plans for the future also involve integrating Prospero with additional indexing services including *WAIS* [4], and once they are deployed, semantic file systems [5], and distributed indices [6]. This will be accomplished by allowing a Prospero server to make meta-information from these databases available using the Prospero protocol. In many respects, the goals of Prospero are similar to those of systems such as *World Wide Web* [7] and *Gopher* [8]. We hope to make information from those systems available through Prospero.

We also plan to add support for organizing resources that are available only on paper. Indices for such information can be made available by running a Prospero server over a bibliographic database. The references would indicate the information needed to obtain a copy of the document, either an ISBN number or perhaps the shelf location in the local library.

We plan to implement a new application interface for Prospero in order to allow use by existing applications without relinking. This will be accomplished by adding Prospero support to an NFS server, the same approach taken by semantic file systems [5] and *Alex* [9]. We hope to benefit from changes already made in those systems.

Finally, the Prospero protocol provides a lightweight protocol for querying directories and obtaining file attributes. We encourage its use as a base upon which other systems can be built.

Obtaining Prospero

A prototype is available and has been used to organize information on Internet sites world-wide. The prototype allows users to construct virtual systems and to navigate through them. Programs linked with the Prospero compatibility library are able to specify file names relative to the active virtual system when opening files. At present the standard release does not support filters. Filter support is available on request, but only for particular architectures.

The release includes the client programs and the Prospero library. A server is also provided, and all sites supporting anonymous FTP are encouraged to run a Prospero server. In addition to the basic release, there are several standalone applications that rely on Prospero to retrieve directory information from *archie*. For more information, or to learn how to obtain the release, please contact:

B. Clifford Neuman
USC-Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
+1 310-822-1511 • info-prospero@isi.edu

Summary

Recent growth of the Internet has increased the amount of information that is accessible and the number of resources that are available to users. Customization allows users to better organize the information that is of interest to them.

Prospero is a distributed directory service and file system that allows users to construct customized views of these resources while taking advantage of the structure imposed by others. Prospero provides the framework within which various indexing services can be tied together, producing the fabric upon which other resource discovery techniques can operate.

Acknowledgments

Ed Lazowska provided valuable guidance throughout this work. Discussions with John Zahorjan, Hank Levy, and Alfred Spector helped me refine the ideas that ultimately led to the development of Prospero. George Ferguson, Khun Yee Fung, and Brendan Kehoe, wrote and ported Prospero-based clients for *archie.* Celeste Anderson, Ben Britt, Steve Cliffe, Peter Danzig, Peter Deutsch, Alan Emtage, Deborah Estrin, Jon Postel, Ole Jacobsen and Dennis Hollingworth commented on drafts of this article.

References

[1] Benford, S,. "Components of OSI: X.500 Directory Services," *ConneXions,* Volume 3, No. 6, June 1989.

[2] Howard, John H., Kazar, Michael L., Menees, Sherri G., Nichols, David A., Satyanarayanan, M., Sidebotham, Robert N. and West, Michael J., "Scale and Performance in a Distributed File System," *ACM Transactions on Computer Systems,* Volume 6, No. 1, February 1988.

[3] Deutsch, P. & Emtage, A., "The *archie* System: An Internet Electronic Directory Service," *ConneXions,* Volume 6, No. 2, February 1992.

[4] Kahle, Brewster, "An Information System for Corporate Users: Wide Area Information Servers," *ConneXions,* Volume 5, No. 11, November 1991.

[5] Gifford, David K., Jouvelot, Pierre, Sheldon, Mark A., and O'Toole Jr., James W., "Semantic File Systems," *Proceedings of the 13th ACM Symposium on Operating Systems Principles,* October 1991.

[6] Danzig, Peter B., Ahn, Jongsuk, Noll, John, and Obraczka, Katia, "Distributed Indexing: A Scalable Mechanism for Distributed Information Retrieval," *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval,* October 1991.

[7] Berners-Lee, Tim, Cailliau, Robert, Groff, Jean-François and Pollermann, Bernd, "World-Wide Web: The Information Universe," *Electronic Networking: Research, Applications and Policy,* Volume 2, No.1, Spring 1992.

[8] "The Internet Gopher: A Distributed Information Service," *ConneXions,* Volume 5, No. 11, November 1991. (See also p. 10)—>

[9] Cate, Vincent, "Alex: A Global File System," *Proceedings of the Workshop on File Systems,* May 1992.

CLIFFORD NEUMAN is a computer scientist at the Information Sciences Institute of the University of Southern California. He has all but defended his doctoral dissertation at the University of Washington where the work described in this article began. After receiving a S.B. degree from the Massachusetts Institute of Technology in 1985 he spent a year working for Project Athena where he was one of the principal designers of the *Kerberos* authentication system. He continues to work on organization and security in distributed systems. He may be reached as: `bcn@isi.edu`.

The Internet Gopher: A Distributed Server Information System

by Mark McCahill, University of Minnesota

Introduction

The Internet Gopher is a simple client/server protocol designed to be used for building distributed information systems and organizing access to Internet resources. In the year since it was released, Gopher has become quite popular. Gopher is now used at over 100 sites worldwide to distribute information including *news,* genome databases, Q&A about computer systems, and collections of Telnet-accessible library catalogs. Since Gopher is a distributed server protocol, Gopher clients have transparent access to information both locally and at other sites. How is this information presented to the user?

A user's view of Gopher

The design goal for Gopher was to make navigating the Internet and accessing distributed resources easy for naive non-technical users. Because users have different preferences for accessing information, Gopher client software combines seamless browsing across multiple servers and full-text searching functions. Naive users typically do not want to learn how to use a large number of programs, and gateways make services such as *FTP, WAIS, Archie,* and USENET *news* available to Gopher clients without using special software. Because all these systems are accessible through a Gopher client, users are not forced to learn several different pieces of software to access information on different systems. Gopher is best thought of as an extensible framework for organizing and building information systems.

Gopherspace

The Internet Gopher client software presents users with a virtual information hierarchy (*Gopherspace*) that they can navigate by either browsing a hierarchical collection of items, or search by submitting queries to full-text search engines. To browse in Gopherspace, the Gopher client software presents the user with lists of items from which the user selects items of interest (typically by pointing and clicking with a mouse). For instance, a user might start looking for a recipe from the list returned by a Gopher server [Figure 1] by looking in the "fun & games" directory for the "recipes" directory which contains a "seafood" directory. Once in the "seafood" directory, the user might select a recipe found in this listing [Figures 2 and 3].

The user could also search instead of browsing. A user might select a full-text search engine called "Search lots of places at the University of Minnesota" from the list shown in Figure 1. When this item is selected, the user is prompted to enter words for which to search [Figure 4] and a server performs a full-text search. The result of the search is a list of items that matched the search criteria, and the user can open the items of interest. The list returned by the search engine is ordered by relevance so the best matches are at the top of the list.

In the figures, there are small icons which denote the type of each item in the list so that users have a cue about what the client will display if an item is selected. Client software on different platforms use this type information to present the user with a familiar representation of the object. The figures show a Macintosh Gopher client where icons are used to represent the type. On a NeXT machine, a browser-like interface is presented to the client, while the generic VT100 UNIX client uses characters to denote type, and so on.

Bookmarks

Most Gopher clients support *bookmarks.* A bookmark is a way for a user to save the descriptor for an item in a Gopher menu. Once the bookmark has been saved, the item can easily be accessed in the future without traversing the intermediate steps that were required to locate the information initially.

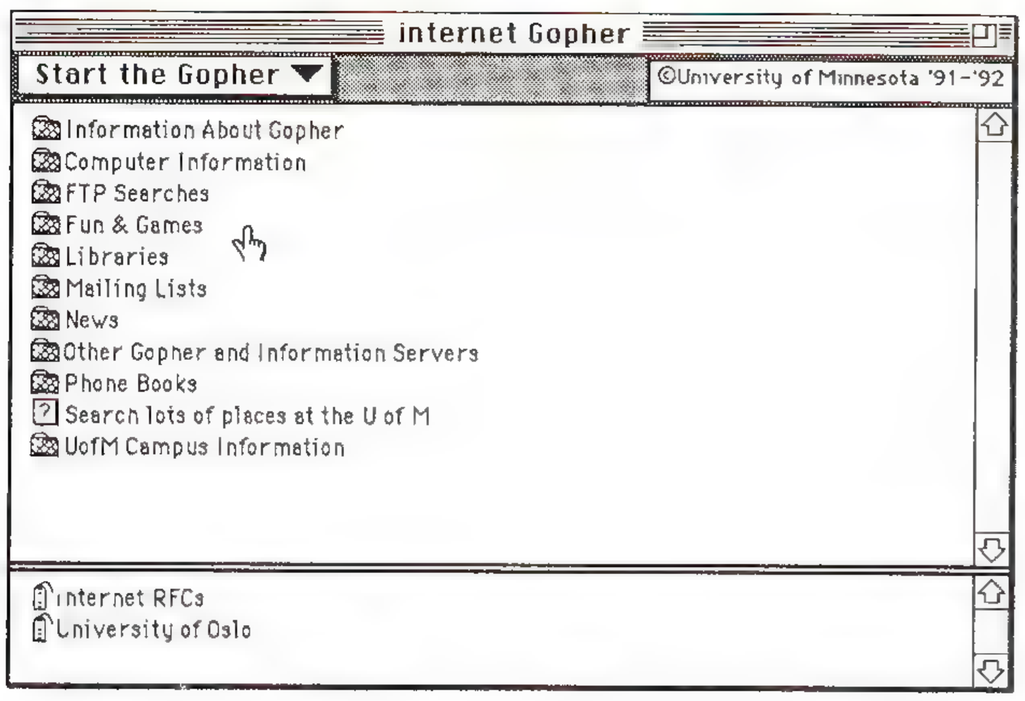


Figure 1

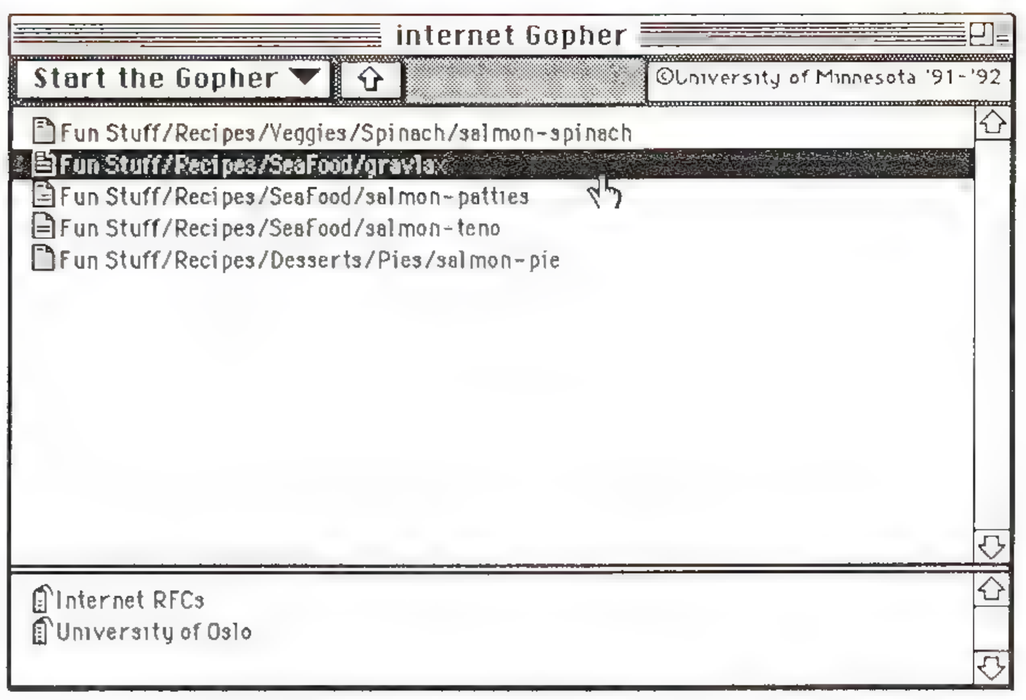


Figure 2

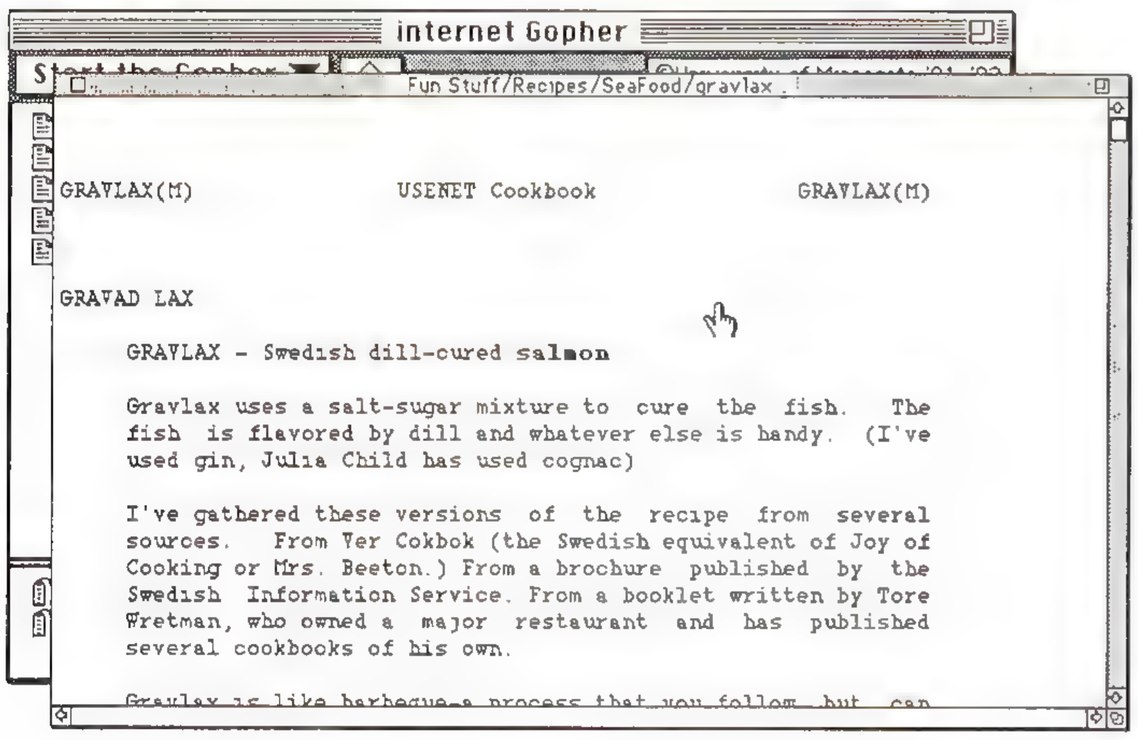


Figure 3

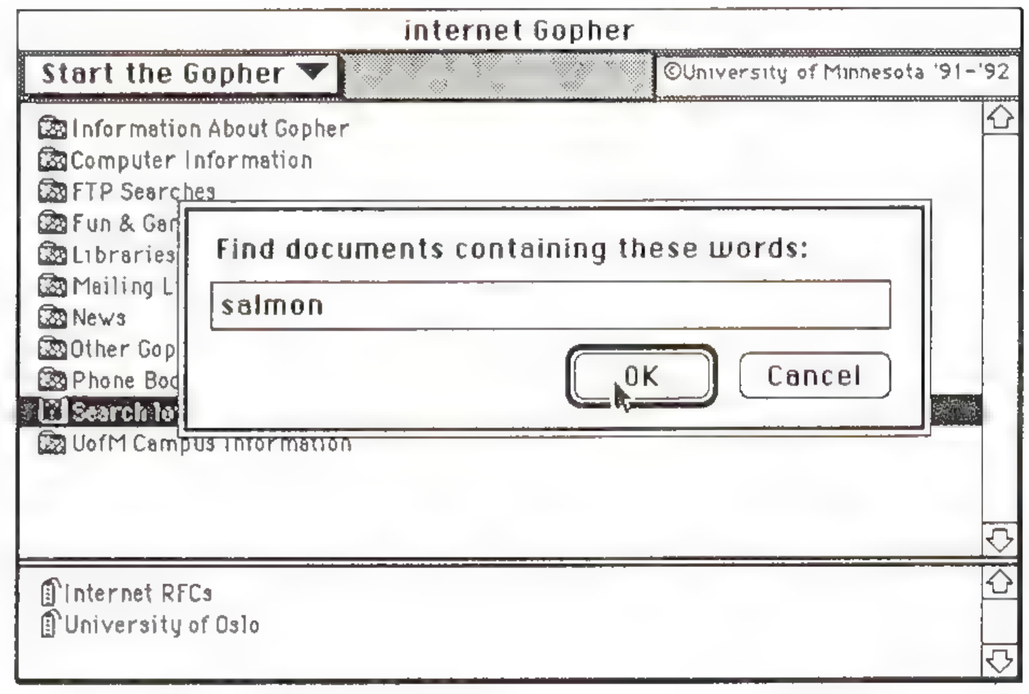


Figure 4

This is especially useful for quickly accessing information buried under several levels of hierarchy and gives each user a way to customize their view of Gopherspace. Two bookmarks (Internet RFCs and University of Oslo) can be seen in Figure 1 at the bottom of the window. To jump directly to the items, the user would select them.

3–D Gopherspace

Most Gopher clients use some form of lists and windows to display the Gopher hierarchy, but work is being done on other user interface metaphors. At Rice University, Steve Ludtke developed a 3-D Gopher client called "Gopher in a Tree" to present the Gopher information hierarchy. This NeXT Gopher client displays each collection of items the user has encountered on a server as part of a 3-dimensional tree growing out of a plane. When the user accesses a piece of information that resides on a different server, a new tree is started on the plane some distance from the first tree with a long branch connecting the two trees. As the user navigates from one server to another, there is a brief animation of flying over the plane from the first tree to the second tree displayed on the screen. In the process of using this client, you gradually build up a tree-filled plane as you explore Gopherspace.

Architecture

One of Gopher's strengths is that information at a server in Italy can easily be referenced by a server in Indiana and these geographically distributed servers are accessed transparently by the user. How is this done?

continued on next page

The Internet Gopher *(continued)*

To begin navigating Gopherspace, the Gopher client software must be configured with the address of a "root" Gopher server. When the client is launched, it opens a TCP connection to the root server and sends a request for an initial list of items to display to the user. The server returns a description of each item that includes:

- The type of the item (text, search engine, etc.)
- The name of the item (to be displayed to the user)
- The selector string (sent to a server to retrieve the item's contents)
- The port and domain name of the machine on which the item resides

The machine name and port can easily be used to refer to information on a different server than the one that returned the list of items. References to information residing on other servers are called *links.* Links to items that reside on other machines can refer to individual documents, search engines, directories (collections of items), or any other Gopher item type.

Client–Server transactions

Since a Gopher client sends a single request to a server, receives the server's response, then closes the connection to the server, the protocol puts minimal load on a server. Once a client has retrieved a list of items from the server, it can access items on other servers without connecting to the first server again. In fact, most clients cache the lists of items they receive from servers to improve performance.

The virtues of this architecture are clear when you look at the root level server for the University of Minnesota. This server has handled a 24-hour average of one transaction every 15 seconds for the last five months. The server hardware is a Mac IIfx running Apple's A/UX UNIX operating system. In spite of the continuous transaction traffic, this server has very low system load averages.

Moreover, if a server is getting heavily loaded, some of its information can be moved to another server and links to that information placed on the first server. This makes it possible to change where information resides without affecting the user's view of the organization of Gopherspace.

Another virtue of links is that a server that acts as a search engine can return links to items that reside on other Gopher servers. It is possible to build and search full-text indexes on faster workstations, for information that is served from slower (and cheaper) systems.

Links to items on other Gopher servers also make it possible for each site to organize the hierarchy of information to suit their needs and still provide access to information at other sites. For instance, most users at the University of Minnesota configure their Gopher clients to connect to a Minnesota root Gopher server, while users in Oslo connect to their own local root server. Since both the server in Minnesota and the server in Oslo have an entry with links to other Gopher servers, users can navigate between servers.

Types of information served by Gopher

Because each item has a type associated with it, the client software can differentiate between documents, directories, search engines, sounds, etc. and present the different types appropriately. The type descriptor also makes it easy to add functionality to the Gopher protocol by defining new types for new services.

Currently there are types for:

- Directories
- Text files
- Search engines
- CSO/PH servers (an electronic phonebook database)
- Descriptors for Telnet sessions (to access terminal-based information)
- Digitized sounds
- Binary files

Search engines

When a user selects a Gopher search engine, the client software takes the words that the user wants to search on and sends this string to the server. The search engines accepts the string and returns a list of items to the client. The list of items includes a numeric score for the relevance of each the items to the search criteria, so it is possible for the client to rank the items returned.

In the first release of the Gopher software, only NeXT computers could be full-text search engines since Gopher depended on NeXT's *Digital Librarian* software for building and searching full-text indexes. Later releases of Gopher server software incorporated the search engine used in the public domain UNIX implementation of WAIS. This made it possible for a UNIX server to have one full-text index for a collection of data and run both a Gopher and a WAIS process to make the information available to both worlds. Recently, we released a Gopher-to-WAIS gateway that makes any WAIS server accessible to Gopher clients.

Although the index for the full-text search generally resides on the server that accepts the query, it is sometimes useful to set up a *fanout* search engine. The fanout search engine accepts queries and then submits them to a predefined list of search engines. The fanout server then gathers up the responses and returns them to the client that initiated the request. A fanout server is used to implement the "Search lots of places at the University of Minnesota" search engine. The fanout search engine simply forwards the search string to all other search engines at the University. Fanout search engines are also useful for taking advantage of the parallelism inherent in a network of servers and for spiting large search databases over several machines.

Gateways

While Gopher can be extended to encompass new types of services, it is often possible to map a service onto an existing Gopher type by using a software gateway to translate from Gopher protocol to another protocol. Gateways from Gopher to other services give Gopher clients seamless access to WAIS, Archie, and FTP sites.

WAIS servers support full-text searches and use a dialect the Z39.50 protocol to communicate with WAIS clients. A Gopher to WAIS gateway translates queries from the Internet Gopher protocol to the WAIS dialect of Z39.50 and gives Gopher users access to all of the WAIS servers without learning how to use anything other than Gopher. The WAIS servers appear to Gopher users as simple search engines.

continued on next page

The Internet Gopher *(continued)*

A similar strategy is used to make *archie* and FTP services accessible to Gopher clients. *Archie* servers maintain a database of files at anonymous FTP sites. With *archie* a user can search the database for file names that contains one or more words, and *archie* returns a list of FTP sites with files that match the search criteria. A gateway maps *archie* services to a Gopher full-text search engine type and reformats *archie* responses so that the Gopher client receives a list of items. The descriptor for these items contains the machine name of a Gopher-to-FTP gateway so that a user can seamlessly fetch the files they located with *archie* from a standard Gopher client.

How Gopher is used

At the University of Minnesota, Gopher is used as a campus-wide information system to electronically publish information such as the campus newspaper, university policies, the campus directory, etc. Gopher is also used for computer support. The University of Minnesota maintains a full-text searchable Q&A database of over 7000 items about microcomputers and workstations. In fact, the original goal of Gopher was to partially automate end user microcomputer support. After putting the Gopher system in place, we have seen a decrease in the volume of phone-in computer support questions.

Other information available in the Gopher system includes genome and biological databases at several different sites, texts of classics such as *Alice in Wonderland* and the CIA *World Factbook,* weather, USENET *news,* and campus wide information systems at other institutions.

Software, documentation and news

The Internet Gopher protocol specification, documentation and software (clients: Macintosh, PC, NeXT, X-windows, VMS, VM/CMS, and VT100 terminal) (servers: UNIX, NeXT, Macintosh, VMS, VM/CMS, and MVS) are available for anonymous FTP from host `boombox.micro.umn.edu` in the `/pub/gopher` directory. Gopher is discussed on the `gopher-news` mailing list (subscription requests to `gopher-news-request@boombox.micro.umn.edu`) as well as on the USENET newsgroup `alt.gopher`.

For a quick look at Gopherspace, you can Telnet to Internet host `consultant.micro.umn.edu` and log in as "`gopher`." The `gopher` account on `consultant.micro.umn.edu` provides access to a VT100 Gopher client. For extended visits to Gopherspace you will probably want to run a Gopher client on your own system; the quality of your Gopher experience will be greatly enhanced by using a mouse.

References

[1] Kahle, Brewster, "An Information System for Corporate Users: Wide Area Information Servers," *ConneXions,* Volume 5, No. 11, November, 1991.

[2] Deutsch, P. & Emtage, A., "The *archie* System: An Internet Electronic Directory Service," *ConneXions,* Volume 6, No. 2, February 1992.

[3] Berners-Lee, T., "A Summary of the WorldWideWeb System," *ConneXions,* Volume 6, No. 7, July 1992.

MARK McCAHILL has been a project leader and manager of distributed computing and support for the University of Minnesota's Computer and Information Systems Department for the last six years. At the University of Minnesota, he has worked on projects to develop easy-to-use network applications such as POPmail for the Mac and PC (SMTP mail client software), electronic directories, and campus backbone network architecture. He is the project leader for the Internet Gopher system and is currently working at providing network access to off-campus computer users. He can be reached via e-mail as: `mpm@boombox.micro.umn.edu`.

INTEROP 92 Spring Proves a Capital Idea

by Daniel P. Dern

21,000+ people converged on the Washington, D.C. Convention Center in mid-May for INTEROP 92 Spring, the first full-scale INTEROP to be held on the East Coast. 255 exhibitors filled 82,900 square feet. (For comparison, INTEROP 91 Fall was 97,500 square feet—and INTEROP 92 Fall currently has 355 exhibitors booking 167,000 square feet.) By all reports, the show went well for exhibitors and attendees alike.

Smoothly

"This is one of the best managed trade shows I've been to, and I've been to a lot lately," stated Steve Roberts, the High-Tech Nomad, who rode BEHEMOTH (*Big Electronic Human-Energized Machine...Only Too Heavy*), his highly computerized and networked recumbent bicycle into town for the event. "I've been impressed how smoothly everything is running, and how nice people are. And everyone here is very literate—even the 'level one' questions come from a perspective of technical excellence."

Roberts' view was echoed almost universally by vendors and attendees, including INTEROP regulars and first-timers, who packed many of the tutorials and sessions, and kept exhibition booth personnel busy.

"We came to the show for two basic reasons: to increase 'face-to-face' time with end users, and to make our presence felt with resellers," stated first-time exhibitor Jeff Shapiro, Product Manager, IMC Networks Corp (Irvine, CA). "This show is a little bit surprising in terms of the number and quality of resellers attending who are looking to further their knowledge."

Packed

"It's great," reported Carl Klessig, President of Enterprise Solutions (West Lake Village, CA), who has several INTEROPs under his belt. "I'm amazed at the amount of attendance. It's been much stronger than I expected. Yesterday we were packed all the time, with seriously interested people—not just a lot of government traffic, but also a fair amount of commercial traffic."

"The caliber of people we've seen here is very high," agreed Ben Littauer of Baranof Software (Brighton, MA). "They're qualified, they understand what we've got and know whether or not it's useful to them. This is very different from most other shows."

INTEROP 92 Spring "dressed up the Convention Center like never before in history," relays a convention industry observer, turning the main entrance to the show floor into a star-filled tunnel.

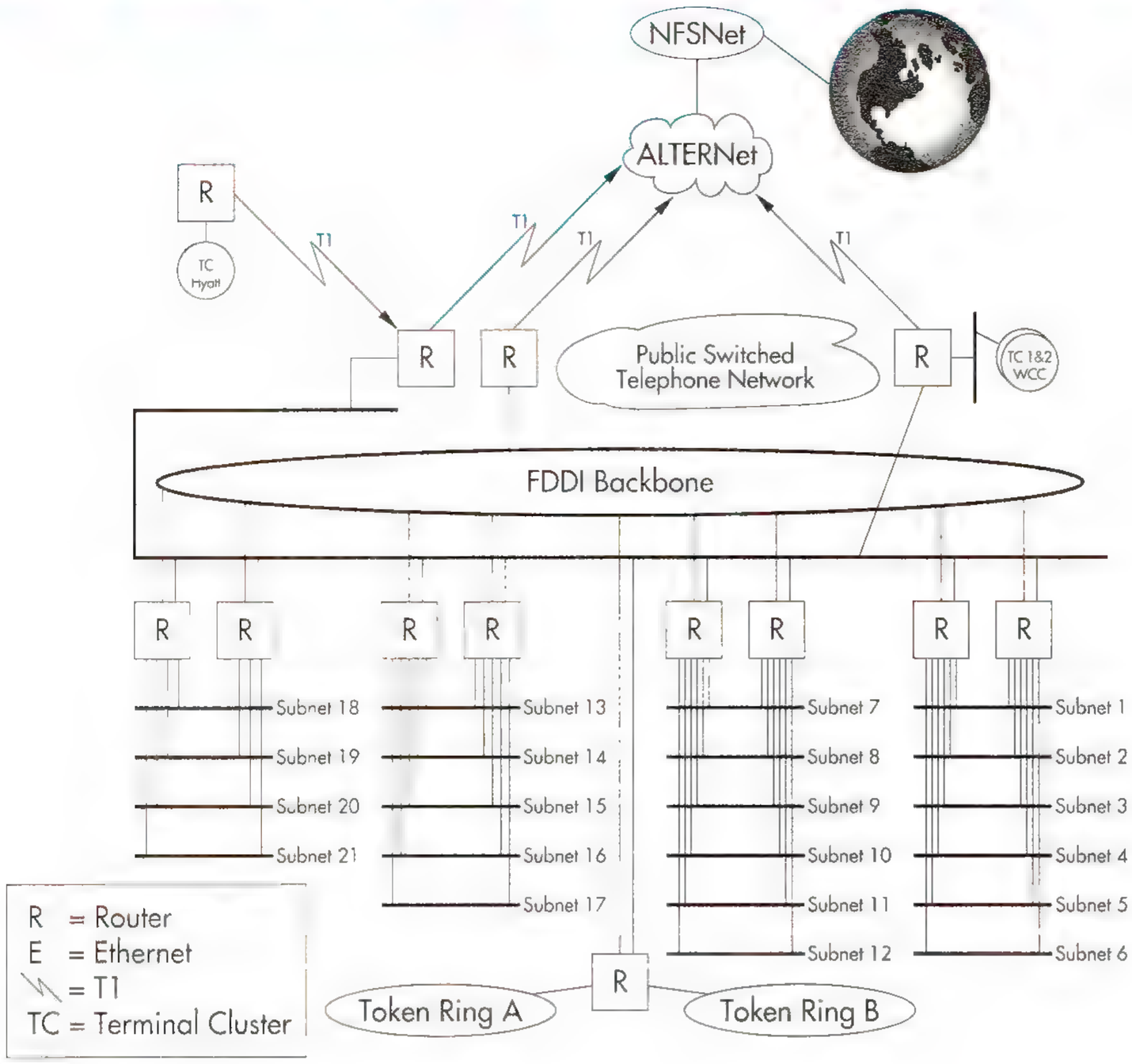
Highlights

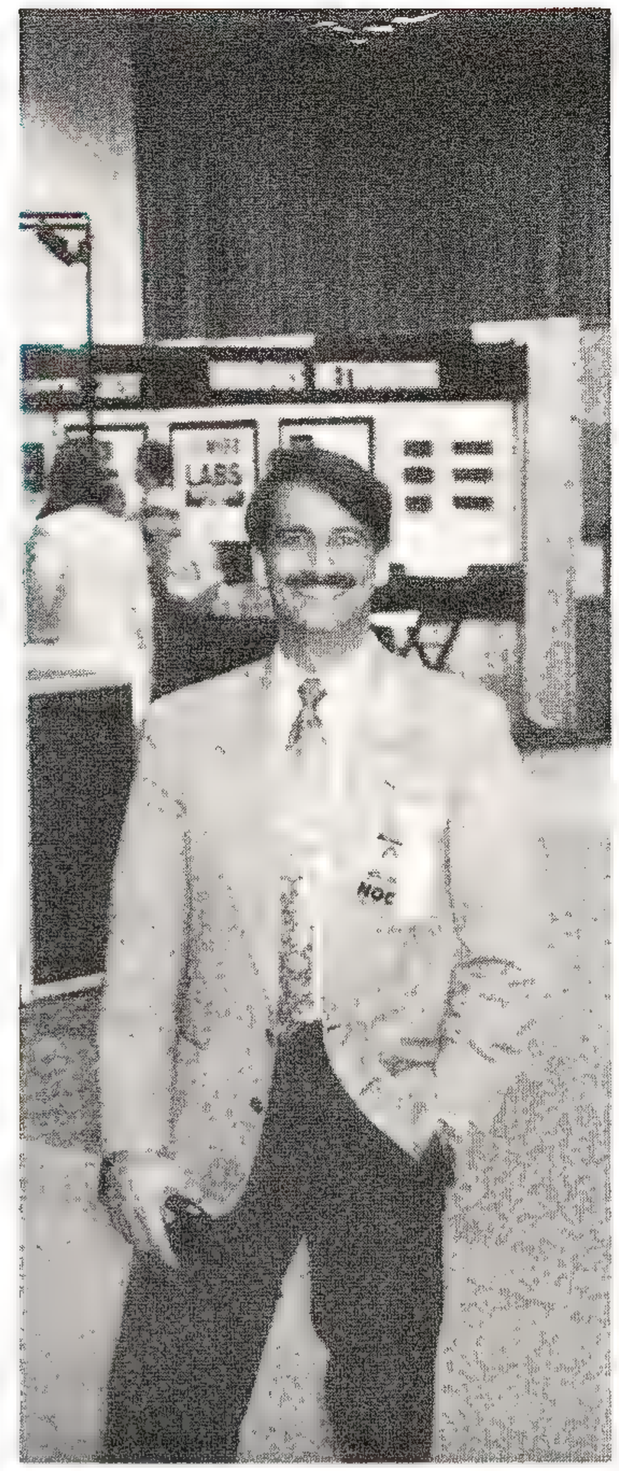
Highlights on this show included:

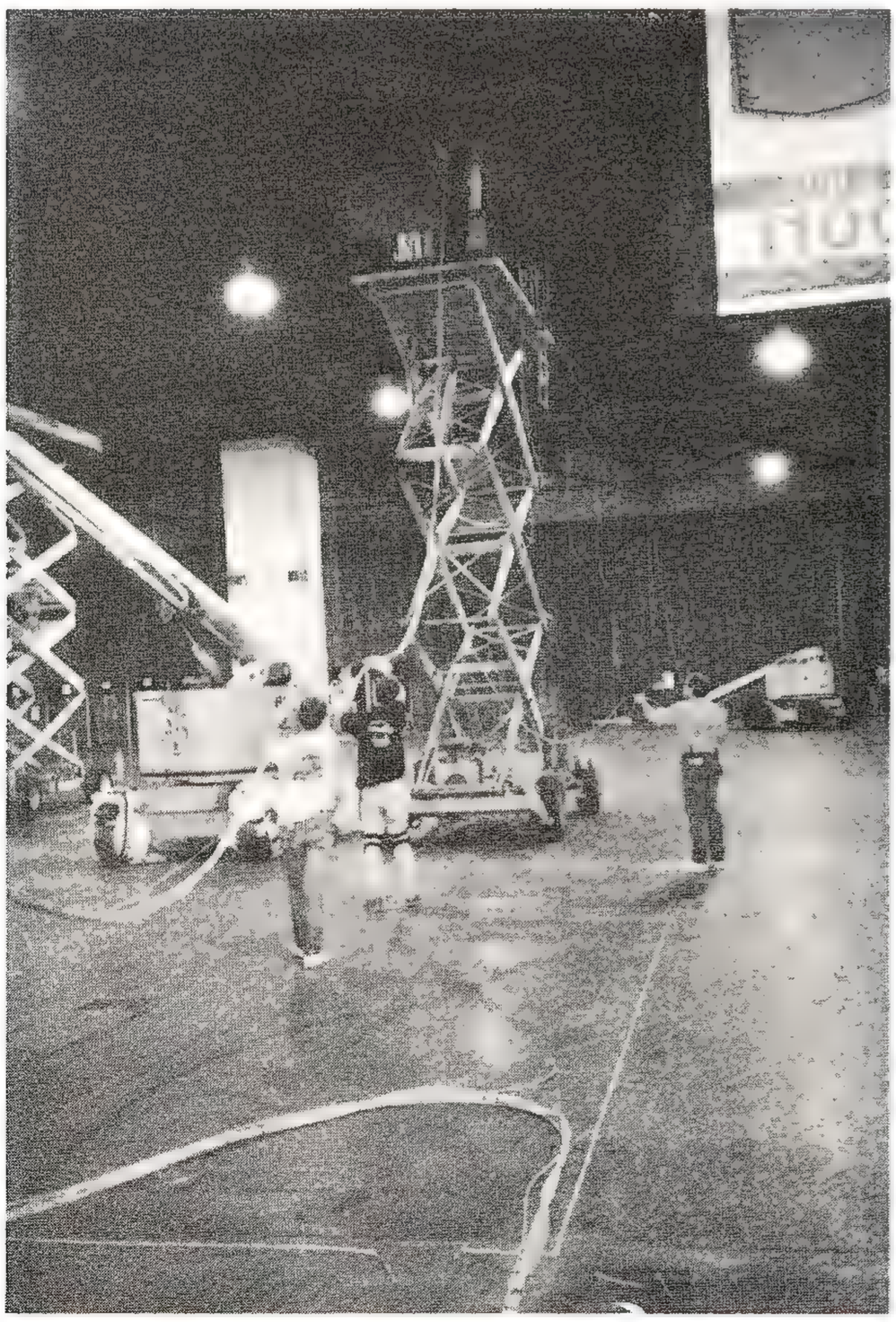
- Eight Solutions Showcase demonstrations: AppleTalk, FDDI, Frame Relay, Network Management, SMDS, SNA–DECnet–TCP/IP Connectivity, Token Ring, and Wireless,
- *The Great OSI Debate,* drawing a full house for "ideameisters" Marshall Rose, Dick desJardins, and Christian Huitema,
- Van Jacobson receiving the *INTEROP Genesis Award,* for his work on network performance improvements,
- Plenary addresses by Mitchell Kapor, President of the Electronic Frontier Foundation, and Dr. Dixon Doll, Accel Partners/DMW Group,

continued on page 18

The INTEROP 92 Spring ShowNet™ Topology

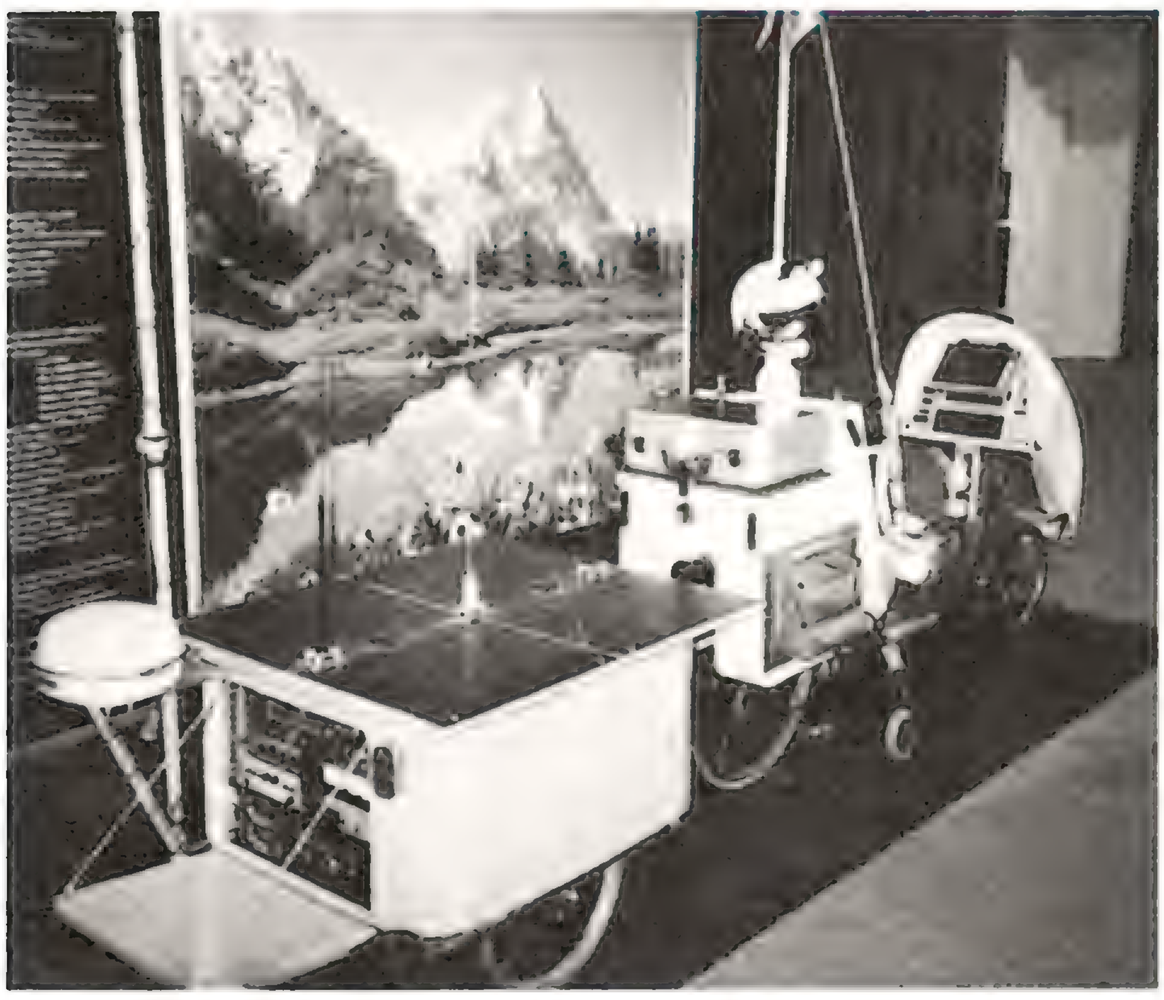






The INTEROP ShowNet comes together through the efforts of a large team of volunteers under the leadership of Michael Tharenos from Interop Company (above). Bundles of cable are "flown" above the show floor with scissor lifts and tied to the building steel 40 feet above.

Above: "The finished product." At the end of each network "rib" are cabinets with various network equipment (far right in photo). Below: Not your average bicycle. Steve Roberts explains BEHE-MOTH to Vint Cerf.



INTEROP 92 Spring *(continued from page 15)*

- Special guest speaker General "Stormin' Norman" Schwarzkopf, on "Desert Storm: Lessons Learned for All of Us." "He was funnier and better than I had imagined," said one observer.
- Local exchanges companies (LECs) Ameritech, Bell Atlantic, BellSouth, NYNEX and other companies announced plans for individual deployment of SMDS and Frame Relay PVC services across the U.S., e.g., for LAN interconnection.
- Interop Company announcing that they're taking the show even further East, and will host *INTEROP 93 Europe* in Paris, October 25–29, 1993.

The ShowNet

ShowNet, the building-wide network that links the equipment in vendor booths to each other and through external links into the Internet, went up smoothly, quickly and painlessly, according to reports, to carry a mix of IP, OSI, IPX and AppleTalk packets over a mix of 10Base-T, Ethernet, Token Ring, FDDI and T1/T3 connections.

Over a hundred ShowNet volunteers came from around the world, including as far as Taiwan, consuming an estimated 4,200 soft drinks and 100 pounds of chocolate as they deployed 61.5 miles of UTP, 2.8 miles of fiber, 1.9 miles of STP, 260 drops, 6,000 connectors, 15,000 cable ties, and 1,200 pieces of equipment.

Even the PR people were happy. Joann Anderson from the Boston office of Copithorne & Bellows observed, "It was one of the most well run shows I've ever attended; [INTEROP] clearly did a lot of good up front planning. All the top networking press was there, all the pubs were represented—including many who made the effort to come up from ICA in Atlanta. Our clients got a lot of exposure and good quality leads."

"My original fears about having two INTEROPs a year were unfounded," confessed Jeff 'That Dawg Can Hunt' Case, SNMP Research, who had been concerned that having more than one INTEROP per year would dilute the attendance. He credits the tutorials with making INTEROP different. "The quality of questions in our two-day class was quite high."

Fun

No INTEROP would be complete without a touch of techno-whimsy, and INTEROP 92 Spring was no exception. At SNMP Research, Jeff Case brought along a Secure SNMP Soda Machine, and over at Epilogue Technology, data from a nearby PC weather card (with both local and "very remote" [Australia] inputs) ran, via network and voice synthesis, to the talking Internet Weather Bear.

"I'm learning a lot about networking," said Steve Roberts, who sandwiched in interviews with the BBC and NPR's Noah Adams. "I've gotten a lot of good ideas from the people here." Perhaps next INTEROP we'll see a Bicycle MIB?

A frequent contributor to *ConneXions,* **DANIEL P. DERN** is a Watertown, Mass. based writer specializing in technology, science and business. He is writing *The Internet Guide for New Users* for McGraw-Hill. (E-mail: ddern@world.std.com)

Opinion: How to Win the Battle of Network Management

by Michael J. Edelman, Sequent Computer Systems, Inc.

Introduction

It seems to be a given that in order to have a well-managed network these days you need to have an NMS (*Network Management Station*) product to assist you. You would think that with the myriad of NMS products available today that we all would have well-managed networks. This is not the case; well-managed networks are the exception rather than the rule.

This conclusion was reached by evaluating many NMS products over the last 18 months. I do not want to imply that all of the NMS products evaluated fail in all areas, rather all NMS products evaluated fail in at least one area.

It appears that many vendors of NMS products missed the target by developing "bells and whistles" and "glitzy" interfaces, sacrificing some of the basic components of network management. It has reached the point that the basic components of network management are either missing entirely or the implementation of the basic components of network management is in some way flawed.

One popular NMS product is being used by many for tasks that the product was not originally designed to do. One problem with this product is that it would create one UNIX process for each device interface being "polled." In our case, that was well over 300 processes; enough to bury our SPARCstations. I also use the term "polled" loosely because if a device failed to respond to one poll, the polling process for that device would hang. These particular flaws in this product have been fixed but the overall model for the product has not changed and the product is recognizably not suited to many of the tasks for which it is being used. It is worth noting that the vendor has recognized the product's shortcomings and plans to solve these problems.

What is network management?

Before you can do anything else with your network, you need to monitor it. This is the "First Basic Need" of network management. Monitoring your network allows you to be alerted when key devices change state, that is, they become available or unavailable. The alerts can be used to initiate a trouble ticket. Tracking and logging these alerts provides an audit trail. The historical data in the audit trail can be valuable when making a determination on the kind of repairs, preventive or otherwise, that need to be made.

The key point is that monitoring your network allows your network support role to transition from re-active to pro-active. Without this transition the network support group will never have time to perform management tasks.

The "Second Basic Need" met by managing your network is trouble shooting, which allows you to find and isolate faults. Trouble shooting should also help you with restoring service, including re-routing of network traffic where possible.

The "Third Basic Need" that network management addresses is gathering data on your network. Managing your network requires that you understand the operating characteristics of your network. You need to be able to answer the question "What is normal?" This requires that you gather data on your network that can be used to generate and update baseline information as well as provide backbone, link, and subnet utilization.

continued on next page

How to Win the Battle *(continued)*

The data can be used for a variety of things like setting threshold triggers that generate an alert when data items have exceeded specified values. To ensure that you are getting the most out of your investment you can use the data to tune your network. Capacity planning is another use for the data: when do you need to upgrade (or downgrade) a device or link? And, even if you do not charge for network resources, you need the data for planning purposes. Without data you cannot manage your network.

The "Fourth Basic Need" is integration. There are two basic kinds of integration: exchanging data with other applications and integration at the application level. An example of complete integration would be that when a network problem occurs the NMS product obtains employee name, phone number, and location from a corporate employee database, and then obtains device type, manufacturer, model number, software version, and cable plant information from a corporate logistics database, for each employee and device involved, and sends this information along with a description of the problem to a trouble ticketing system.

Basic paradigms

There are about a dozen key paradigms that most of the current NMS products use. The first paradigm to look at is the user interface. All the NMS products evaluated have a graphical user interface and lack a character-based-terminal interface, which is okay if your network operations center is staffed 24 hours a day, seven days a week. Otherwise you would need to provide your network support personnel with a graphical device for use at home.

All the NMS products support a centralized management model. None support distributed management. Nearly all of them have one database per seat. This makes the product time consuming to configure, expensive to maintain, and multiple stations difficult to synchronize.

The data management model employed by nearly all the NMS products evaluated is egocentric, that is the NMS product is the center of the data universe. A few NMS products use a flat file internal database; others use an embedded RDBMS internal database; most have a home grown internal database. Many NMS products have to be taken down to add devices or to reconfigure. Nearly all the NMS products lack the ability to do bulk updates requiring the operator to go through multiple menus and pop-ups to add each new device. The few products that allow bulk updates require that the product be taken down to do the bulk update.

None of the products can readily share data with other applications, such as allowing the correlation of logistics or employee information from corporate data sources with devices in the NMS database.

The user interface for most of the NMS products tends to be overly complex. There are too many levels of menus, pop-ups, etc., to traverse in order to accomplish specific, simple tasks. Very few NMS products take advantage of expert system technology. Those that do, with one exception, are of limited use. The one NMS product that has an expert system performed very well, correctly intuiting and differentiating between the sources and symptoms of problems.

A few NMS products require the installation of devices on the subnets you want monitored at a substantial cost (approximately $5000.00 for each subnet monitored). This makes no sense as more networks utilize intelligent hubs and routers, devices that already gather all the data you need.

Another key problem that most NMS products have is in the area of reporting. Most NMS products have very limited, inflexible, reporting tools. Some NMS products will not allow historical data to be reported even though the product generates the file that contains the data. This renders historical data useless since it is stored in an unspecified binary format.

Many of the NMS products lack the ability to either be integrated with other tools and products, or to integrate other tools and products. For example, very few applications will look up an IP address or name using Domain Name Service; all names and addresses have to be specified. A number of tools like *netstat* or *traceroute* are not available from within the product.

Jack of all trades

The bottom line is that an operator using the typical NMS package is required to be an expert at:

- Networking (IP, OSI, Novell, etc.)
- Sun Hardware (typical NMS platform)
- SunOS and UNIX
- SNMP
- MIBs
- DNS and NIS (YP)
- X11 and OpenWindows
- Telecommunications
- Customer Service

What do we need to manage a network?

The first thing we want is to have the NMS package operators be experts at customer service. They should then have enough ability in other areas to be able to determine who to contact to resolve problems (for those problems that are not simple enough to resolve at the NMS console).

We need NMS products to be able to both integrate other tools and products and to be integrated by other tools and products via a standardized API (Application Program Interface). You should be able to enter the name or IP address for one interface of a device and the NMS product should use that information to look up the names and addresses of any other interfaces via DNS, NIS (YP), or SNMP. It should be possible to run tools like *netstat* or *traceroute* without going outside the NMS product. There are many other existing tools out there that should be integrable as well. With a standard API it is straight forward to integrate tools like protocol analyzers, trouble ticketing systems, domain name service, vendor supplied applications, OS applications, and proprietary management tools and applications into a network management system. The ability to integrate tools simplifies and reduces the number of tasks required to implement a network operations center since tools from different sources could be integrated with a seamless appearance.

Views

Most NMS products provide a logical view of your network; others provide a second, geographical, view. The ability to have multiple customizable views is very important. The views and the relationship of the devices within a view should be definable by the administrator that is configuring the system.

continued on next page

How to Win the Battle *(continued)*

For example, a view by application that displays all the devices that need to be available for that particular application to successfully execute, and other views that group devices by device type or owner would also be useful.

Customizable, flexible reporting is another must. Too many NMS products have taken it upon themselves to determine what data you need, how often you need it, and how it should be displayed. The NMS administrator should be the one to determine what data to collect and how often it should be collected. NMS products should allow data to be extracted or processed in either tabular (where it may be fed into a spread sheet) or graphical format. They should not limit the number of data items extracted or plotted.

NMS products need to move away from their egocentric data management practices. The NMS product is not the center of the universe; it is a small part of a bigger picture. NMS products should be able to exchange data with other applications via reading or writing flat files or by sending queries and updates to an RDBMS. There are many other corporate applications that contain data that the NMS product needs to read or write. The NMS product should do so with the proper facilities just like any other corporate application.

Smarts

Very few NMS products include an expert system. Expert systems are extremely useful in network management, particularly an expert system that can be extended (is capable of "learning").

Alerts should be customizable. The alert process in an NMS product should be able to ring the bell, send an arbitrary file to an audio device, produce visual changes (colors, pop-ups, icons, etc.), or send input to an arbitrary program.

The NMS product should have a comprehensive help system and an adaptable user interface, one that both novices and experts can use. The NMS product should use standard mechanisms for customization of the user interface. For example, X11-based NMS products should use X11 "resources" for configuration.

Client–Server Model

NMS products must fit the client/server model. There should be one server process that gathers all the data for a group of devices. The group may be your whole network or a subset of your network. Multiple clients may connect to this server to manage that group of devices. The server process should have the ability to share bulk data with other server processes. This allows the network management responsibility to be divided among multiple servers, for example, by geographic region. Each region should still see all the devices in other regions without having to multiply poll devices. There also needs to be a designated "master" server process that is ultimately in charge, but the "master" designation should be able to "float." For example, the master could move as different parts of the world move into "prime time."

Along with the client/server model there needs to be the ability to have hierarchical levels of access to the NMS product with permissions for each specified level of access. This allows you to open up access to the NMS to all those who need such access without having to worry about compromising security or the data integrity of the NMS.

This brings us to distributed management, allowing different regions or other logical groups to manage their part of the network while still providing a view of the whole. This is absolutely critical for managing a global enterprise.

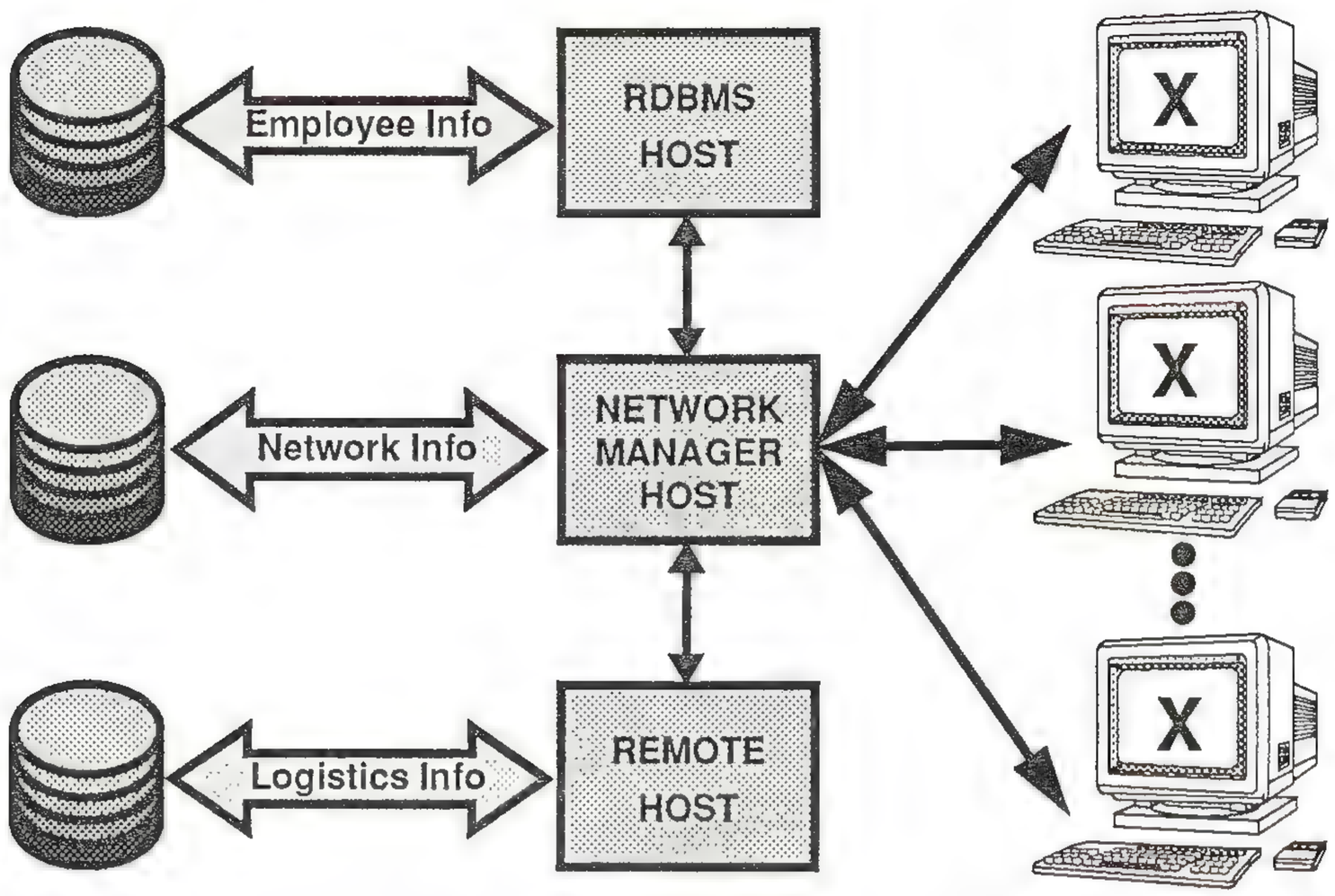


Figure 1: Network Management Paradigm

Figure 1 shows a model for network management, where the network manager exchanges logistics data via flat files with a remote host, and exchanges employee information via queries and updates to an RDBMS on another host. Multiple X clients access the single management station.

What is wrong with this picture?

There are some major problems with the model for network management depicted in Figure 1. Most clients do not know (or care about) the differences between e-mail (or any other favorite application), host systems, and networks. When any one of these fails, the result is the same: clients are unable to get their work done.

As technology advances, networks are becoming very reliable. IP routers, subnetting, intelligent hubs, etc., contribute directly to the reliability of today's networks. Other components (hosts or applications) are often more likely to be the cause of a failure. The solution to this problem requires that we manage more than the network.

Applications must be instrumented so that they are monitorable and manageable via SNMP. Host systems must be monitorable and manageable via SNMP; many are monitorable today; very few, if any, are manageable.

Other hardware should be monitored and managed via SNMP. For example, stand-alone peripherals (e.g., printers), CSU/DSUs, phone switches, multiplexors, UPS systems, environmental control systems, etc., should be monitored and managed via SNMP.

Common sense

This seems like a "common sense" need to me, yet it is not a common capability in the Open Systems environment. Everyone needs their mission critical systems to be up and available. Mission critical systems are no longer confined to just the computers, but also must include the network and the applications.

continued on next page

How to Win the Battle *(continued)*

Why are these capabilities not available? The answer is that they are available with proprietary systems, and have been for some time. Open Systems solutions are almost here (maybe we will see some examples at INTEROP this Fall).

Figure 2 is a model for enterprise management. The manager (which while logically shown as one host could be many) exchanges logistics, performance, etc., data via flat files with several remote hosts. The manager also exchanges asset, employee, accounting, etc., information via queries and updates to RDBMSs on several remote hosts. Multiple X clients access the manager.

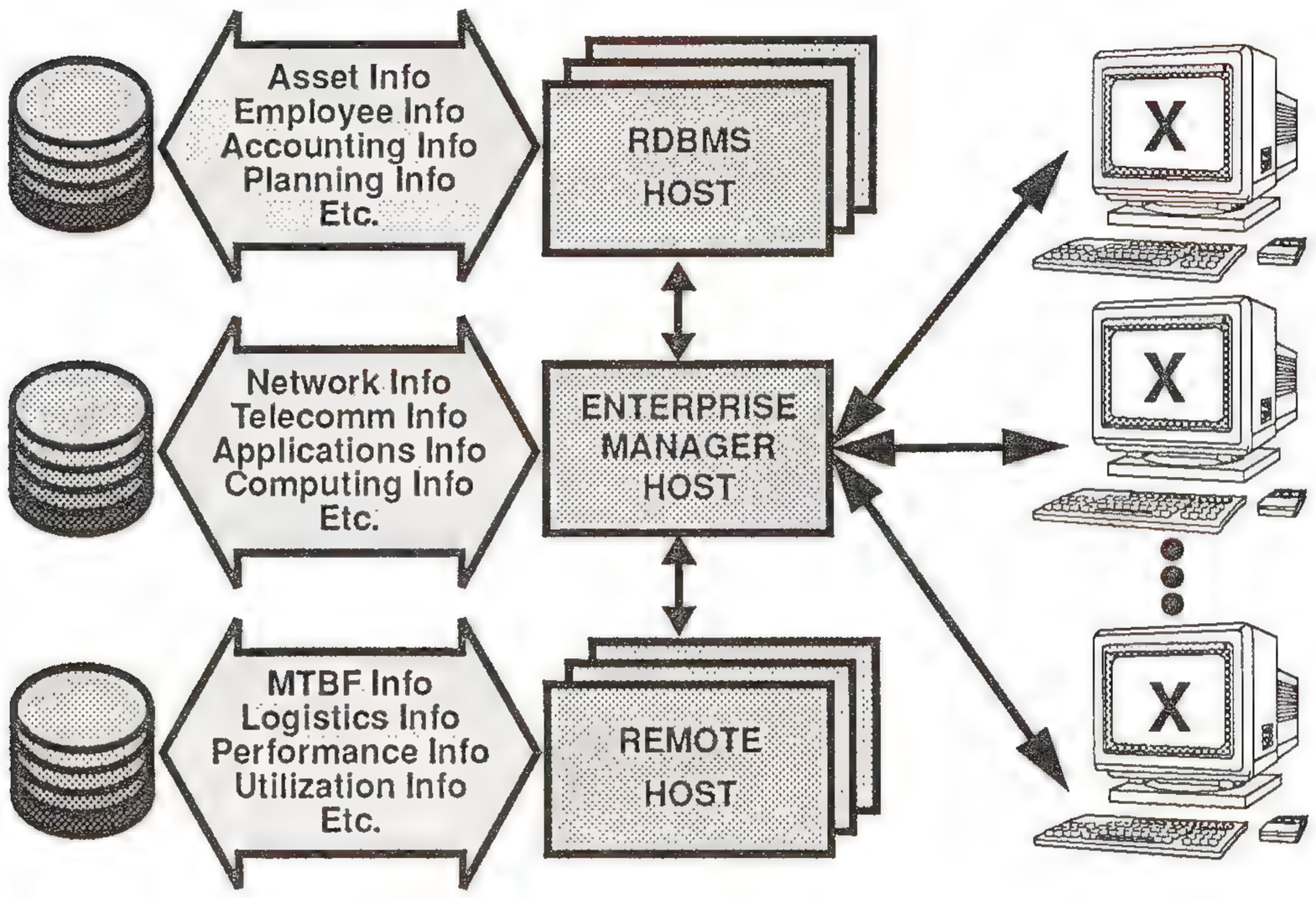


Figure 2: Enterprise Management Paradigm

Where does this leave us today? We have a large number of NMS products that have missed the primary targets of network management. They do not perform the basic management tasks well, if at all.

The larger picture

As a client base, we also missed the target by focusing on network management as the one solution to all our problems. It turns out that network management is only one part of a much bigger problem. One that requires a much more comprehensive solution. This distraction on our part has further confused and diffused the efforts of the NMS vendors.

Instead of fixing the smaller problem of network management, let us focus on the real problem of enterprise management. Fixing the enterprise management problems will fix the problems of application, computing, and network management. The result should be clean, comprehensive solutions that provide vertical and horizontal integration with other tools.

What do we need to manage an enterprise?

To manage an enterprise, we need standards like SNMP, SMI, MIBs, ISO standards, IEEE standards, and Internet RFCs. We need cooperation among all facets of the industry both in driving the standards processes and in adhering to the results. We also need to put more emphasis on the quality of our standards-based products.

It is very important that these products perform as (or better than) advertised. This is the foundation of Open Systems and the Open Systems consumer has the most to gain. As the cumulative R&D dollars spent by Open Systems providers overtake those spent by proprietary systems vendors, Open Systems consumers will see more and faster technological breakthroughs than will be available to proprietary systems consumers.

What you need to do is work with your vendors. Tell them what you need. Tell them what you want. Push them towards standards-based solutions. Pursue solutions from vendors that provide standards-based products and solutions. However, do not expect to implement a complete solution all at once. Take small, safe, steady steps in the right direction. Start with managing your networks. Next manage your hosts, and finally, manage your applications.

Managing an enterprise presents us with a tough challenge. Through Open Systems we can achieve the complete and comprehensive solutions that are required to meet, and exceed, that challenge.

References

[1] *ConneXions,* Special Issue: Network Management and Network Security, Volume 4, No. 8, August 1990.

[2] *ConneXions,* Special Issue: Network Management, Volume 3, No. 3, March 1989.

[3] Marshall T. Rose, *The Simple Book—An Introduction to Management of TCP/IP-based internets,* Prentice-Hall, 1990, ISBN 0-13-812611-9.

MICHAEL J. EDELMAN received his BSCS in 1981 from California Polytechnic State University at San Luis Obispo. From 1981 to 1990 he worked for Tektronix, Inc. in Beaverton, Oregon where he worked on computing and networking projects, including the development of a graphical network monitoring tool. For the past two years he has worked for Sequent Computer Systems, Inc. in Beaverton, Oregon on a variety of architectural projects including the design and implementation of Sequent's internal global enterprise network. He is currently working on architectural projects relating to enterprise management, and the integration of the networking, computing, desktop, and data architectures at Sequent. He can be reached via e-mail as michaele@sequent.com.

Write to *ConneXions*!

Have a question about your subscription? Are you moving, and need to give us your new address? Suggestions for topics? Want to write an article? A letter to the Editor? Have a question for an author? Need a *ConneXions* binder? Want to enquire about back issues? (there are now sixty-four to choose from; ask for our free 1987–1992 index booklet). We want to hear from you. Simply write, call, fax, or e-mail us at:

ConneXions—The Interoperability Report
480 San Antonio Road, Suite 100
Mountain View, CA 94040–1219
USA
Phone: +1 415-941-3399 or 1-800-INTEROP (Toll-free in the USA)
Fax: +1 415-949-1779
E-mail: connexions@interop.com

A Summary of the WorldWideWeb System

by Tim Berners-Lee, CERN

Introduction

The WorldWideWeb (WWW) project merges the techniques of information retrieval and hypertext to make an easy but powerful global information system. The project is based on the philosophy that much academic information should be freely available to anyone. It aims to allow information sharing within internationally dispersed teams, and the dissemination of information by support groups. Originally aimed at the High Energy Physics community, it has spread to other areas and attracted much interest in user support, resource discovery and collaborative work areas.

Reader view

The WWW world consists of documents, and links. Indexes are special documents which, rather than being read, may be searched. The result of such a search is another ("virtual") document containing links to the documents found. A simple protocol ("HTTP") is used to allow a browser program to request a keyword search by a remote information server.

The web contains documents in many formats. Those documents which are hypertext, (real or virtual) contain links to other documents, or places within documents. All documents, whether real, virtual or indexes, look similar to the reader and are contained within the same addressing scheme.

To follow a link, a reader clicks with a mouse (or types in a number if he or she has no mouse). To search and index, a reader gives keywords (or other search criteria). These are the only operations necessary to access the entire world of data.

Information provider view

The WWW browsers can access many existing data systems via existing protocols (FTP, NNTP) or via HTTP and a gateway. In this way, the critical mass of data is quickly exceeded, and the increasing use of the system by readers and information suppliers encourage each other.

Making a web is as simple as writing a few SGML files which point to your existing data. Making it public involves running the FTP or HTTP daemon, and making at least one link into your web from another. In fact, any file available by anonymous FTP can be immediately linked into a web. The very small start-up effort is designed to allow small contributions. At the other end of the scale, large information providers may provide an HTTP server with full text or keyword indexing. This may allow access to a large existing database without changing the way that database is managed. Such gateways have already been made into Digital's VMS/Help, Technical University of Graz's "Hyper-G," and Thinking Machine's "WAIS" systems.

The WWW model gets over the frustrating incompatibilities of data formats between suppliers and reader by allowing negotiation of format between a smart browser and a smart server. This should provide a basis for extension into multimedia, and allow those who share application standards to make full use of them across the web.

This summary does not describe the many exciting possibilities opened up by the WWW project, such as efficient document caching, the reduction of redundant out-of-date copies, and the use of knowledge daemons. There is more information in the online project documentation, including some background on hypertext and many technical notes.

Try it!

You can try the simple line mode browser by telnetting to host `info.cern.ch` (no user or password). You can also find out more about WWW in this way. This is the least sophisticated browser—remember that the window-oriented ones are much smarter!

It is much more efficient to install a browser on your own machine. The line mode browser is currently available in source form by anonymous FTP from node `info.cern.ch`[currently 128.141.201.74] as:

`/pub/www/src/WWWLineMode_v.vv.tar.Z`.

(`v.vv` is the version number—take the latest.)

Also available is a hypertext editor for the NeXT workstation (`WWWNeXTStepEditor_v.vv.tar.Z`), the *ViolaWWW*browser for X11, and a skeleton server daemon (`WWWDaemon_v.vv.tar.Z`).

Documentation is readable using WWW. A plain text version of the installation instructions is included in the tar file! Printable (*PostScript*) documentation and articles are in `/pub/www/doc`.

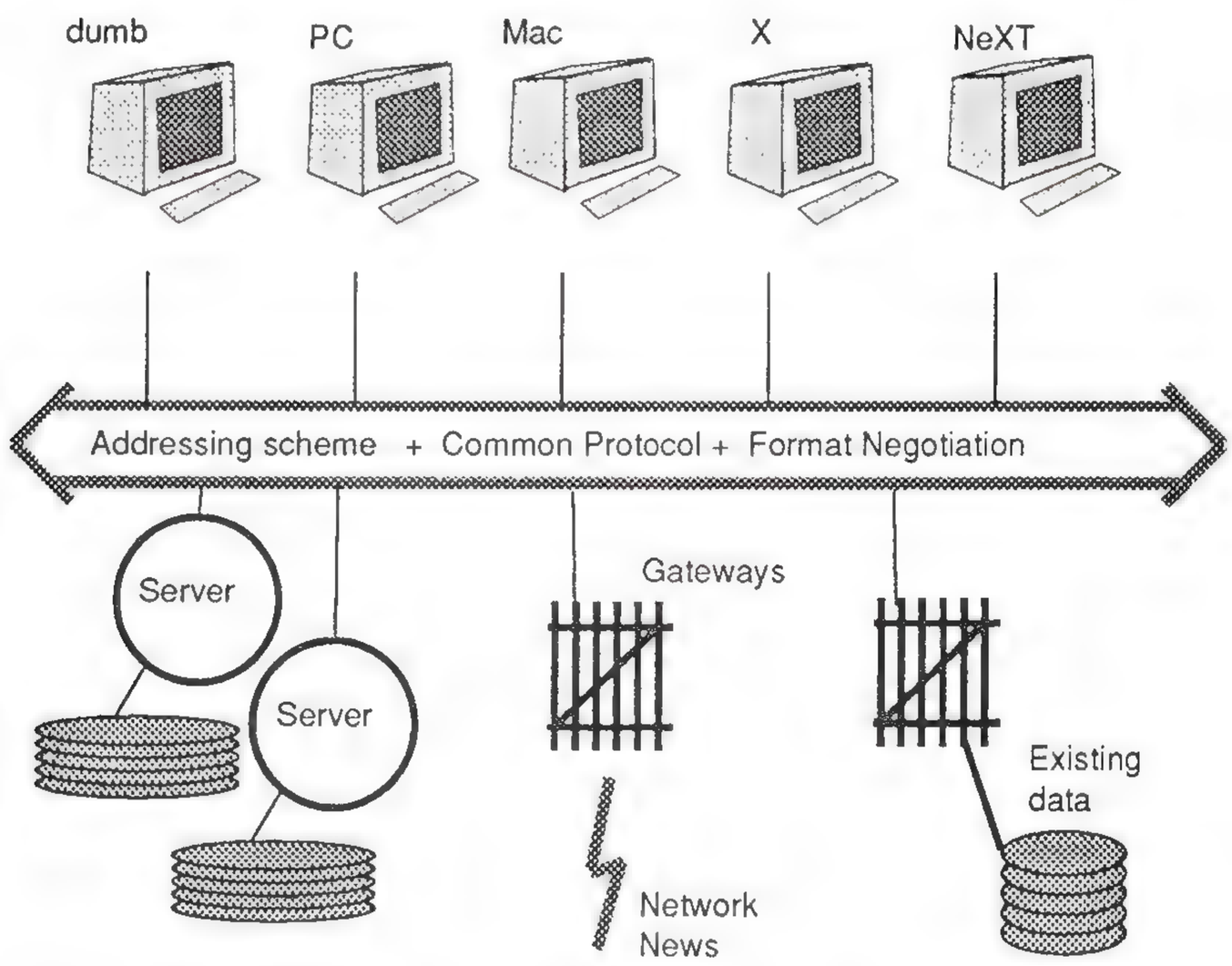


Figure 1: The WWW architecture in outline

References

[1] Barron, Billy, "Another use of the Internet: Libraries Online Catalogs," *ConneXions,* Volume 5, No. 7, July 1991.

[2] Quarterman, John, "Networks: From Technology to Community," *ConneXions,* Volume 5, No. 7, July 1991.

[3] Schwartz, Michael F., "Resource Discovery and Related Research at the University of Colorado," *ConneXions,* Volume 5, No. 5, May 1991.

[4] Kahle, Brewster, "An Information System for Corporate Users: Wide Area Information Servers," *ConneXions,* Volume 5, No. 11, November 1991.

[5] Deutsch, P. & Emtage, A., "The *archie* System: An Internet Electronic Directory Service," *ConneXions,* Volume 6, No. 2, February 1992.

(See also articles on Prospero and the Internet Gopher in this issue.)

Announcement and Call for Papers

The *USENIX Winter 1993 Technical Conference* will be held in San Diego, California, January 25–29, 1993. The conference is subtitled "The Challenge of Innovation." The USENIX Association's twice yearly general conferences are well recognized as leading forums for the communication of new research and the investigation of important technological developments in UNIX or UNIX-inspired computing systems.

About USENIX

USENIX, the UNIX and Advanced Computing Systems Professional and Technical Association, is a not-for-profit, membership organization. USENIX is dedicated to:

- Sharing ideas and experience relevant to UNIX inspired and advanced computing systems,
- Fostering innovation and communicating both research and technological developments,
- Providing a neutral forum for the exercise of critical thought and airing of technical issues.

USENIX achieves these goals through its general conferences and its frequent special topic workshops and symposia. Further, the Association publishes proceedings of its meetings, its bimonthly newsletter *;login:*, and, with the University of California Press, the refereed technical quarterly Computing Systems. USENIX also actively participates in and reports on various ANSI, IEEE and ISO standards efforts.

Important dates

Extended Abstracts Due:	July 20, 1992
Notifications to Authors:	August 19, 1992
Final Papers Due:	November 20, 1992
Preregistration materials available:	Mid-September, 1992
Preregistration savings deadline:	January 4, 1993
Tutorial Program:	January 25–26, 1993
Technical Sessions:	January 27–29, 1993
Birds-of-a-Feather Sessions:	January 26–28, 1993
USENIX Reception:	January 28, 1993 (evening)

Topics

UNIX and its cousins find themselves in increasing use throughout the industry. Succeeding in the challenge of meeting the world's expectations of software is an increasingly difficult task. This USENIX conference is looking for innovative papers in a variety of areas, for example:

- *Computing in the very large:*
  Global connectivity
  Coping with explosive growth
- *Distributed environments:*
  Client–server
  Location independent computing

- *Sociological and societal impacts:*
  Connectivity vs. Security
  New base applications
- *Utilizing state-of-the-art hardware:*
  Storage systems
  Communications systems
  Large networks
- *Exploiting increasingly layered software:*
  Object oriented systems
  Creative new interfaces
  Complexity management
  New development techniques
- *Visionary base systems software:*
  New building blocks
  Novel leverage of standards

At the USENIX Winter 1993 Technical Conference, systems researchers and developers, systems administrators, software professionals, programmers, applications developers, support staff, technical managers, and educators tackle questions of immediate importance to advanced computing systems development and management.

The Program Committee solicits new work on all topics related to UNIX or UNIX-inspired programming and technologies. Vendors are welcome to submit technical presentations, but the program committee will reject product announcements.

Submissions

Submissions must be in the form of extended abstracts (1500–2500 words or 3–5 pages in length). Shorter abstracts might not give the program committee enough information to judge your work fairly and, in most cases, your submission will be rejected. Longer abstracts and full papers simply cannot be read by the committee in the time available. Feel free to append a full paper to an extended abstract; this is sometimes useful during evaluation. The extended abstract should represent your paper in "short form." The committee wants to see that you have a real project, that you are familiar with the work in your area, and that you can clearly explain yourself.

A Good Extended Abstract Should Contain:

- Abstract (same as in the final paper)
- Introduction (to the problem and its importance)
- Solution (details on the problem and its issues, design decisions, tradeoffs, motivations, implementation details)
- Evaluation
- References to previous work

continued on next page

Announcement and Call for Papers *(continued)*

Every Submission Should Include:

- The extended abstract
- One contact author with a daytime phone number and surface mail address
- E-mail address, if available
- Home phone number (volunteers work at night!)
- Indication if any authors are students
- List of audio/visual equipment desired beyond microphone and overhead projector
- Please note: presentations are usually scheduled for 25 minutes

Six paper copies of each submission should be sent to:

Rob Kolstad
7759 Delmonico Drive
Colorado Springs, CO 80919
Phone: +1 719-593-9445
E-mail: kolstad@bsdi.com.

Tutorials

Explore topics essential to successful use and development of UNIX and UNIX-like operating systems, The X Window System, networking and interoperability, advanced programming languages, and related areas of interest. The USENIX Association's well-respected program offers you introductory and advanced, intensive yet practical tutorials. Courses are presented by skilled teachers who are hands-on experts in their topic areas. In San Diego, USENIX will offer tutorials such as:

- Topics in Systems Administration
- Distributed File System Administration
- UNIX Programming Tools
- Systems and Network Security
- Kernel Internals: OSF/1, SVR4, 4.4BSD
- Developing and Debugging X-Based Applications
- Network Program Maintenance and Design
- Introductions to C++ and *Perl*
- Micro-Kernel Technologies
- POSIX Threads and Systems Programming

In an effort to continue to provide the best possible tutorial slate, USENIX is soliciting proposals for new tutorials. If you are interested in presenting a tutorial, contact the Tutorial Coordinator: Daniel V. Klein: +1 412-421-2332 • dvk@usenix.org.

Invited Talks

As part of the technical sessions, a full series of invited talks provide introductory and advanced information about a variety of interesting topics, such as using standard UNIX tools, tackling system administration difficulties, or employing specialized applications. We welcome suggestions for topics as well as request proposals for particular Talks. In your proposal, state the main focus, include a brief outline, and be sure to emphasize why your topic is of general interest to our community.

Work-in-Progress reports These reports provide researchers with 10 minutes to speak on current work and receive valuable feedback. Present your interim results, novel approaches, or newly-completed work. Schedule your report in advance or on-site.

BOFs Birds-of-a-Feather sessions (BOFs) bring together devotees of many varied disciplines for discussions, announcements, mingling, and strategy sharing. Schedule a BOF in advance or on-site

Awards A cash prize will be awarded by the conference program committee for both the best paper and the best paper by a full-time student at the conference. With your submission, please indicate if you are a full-time student.

More information Materials containing all details of the technical and tutorial program, conference registration, hotel and airline discount and reservation information will be mailed in September 1992. If you wish to receive the preregistration materials, please contact:

USENIX Conference Office
22672 Lambert St., Suite 613
El Toro, CA 92630 Phone: +1 714-588-8649 • Fax: +1 714-588-9706

Upcoming Events

SIGCOMM '92—Communications Architectures and Protocols will be held in Baltimore, Maryland, August 17–20, 1992. The conference is sponsored by the ACM Special Interest Group on Data Communications.

Scope The conference provides an international forum for the presentation and discussion of communication network applications and technologies, as well as recent advances and proposals on communication architectures, protocols, algorithms, and performance models.

More information A summary of the conference topics is given below. If you would like more information, please send a (null) e-mail message to the SIGCOMM '92 information server:

```
sigcomm92-prog@ihburn.att.com.
```

An automatic reply will send you a copy of the full advance program with registration forms.

Program Summary

Monday 17 August, Tutorials:

- "ATM/B-ISDN," Bharat Doshi, Subramanyam Dravida, and John Swenson AT&T Bell Labs
- "Optical Networks," Zygmunt Haas, AT&T Bell Labs

Tuesday 18 August:

- Keynote Session
- Protocol Design
- Routing
- Quality of Service

Wednesday 19 August:

- Congestion Control
- Performance Analysis
- Multicast Communications
- Media Access

Thursday 20 August:

- End-to-End Issues
- Network Measurement
- Closing Session

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040
415-941-3399
FAX: 415-949-1779

ADDRESS CORRECTION
REQUESTED

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1